Explicit Nonlinear Model Predictive Control of the Air Path of a Turbocharged Spark-Ignited Engine

Jamil El Hadef, Sorin Olaru, Pedro Rodriguez-Ayerbe, Guillaume Colin, Yann Chamaillard, and Vincent Talon

***Abstract*— Pollutant emissions and fuel economy objectives have led car manufacturers to develop innovative and more sophisticated engine layouts. In order to reduce time-to-market and development costs, recent research has investigated the idea of a quasi-systematic engine control development approach. Model based approaches might not be the only possibility but they are clearly predetermined to considerably reduce test bench tuning work requirements. In this paper, we present the synthesis of a physics-based nonlinear model predictive control law especially designed for powertrain control. A binary search tree is used to ensure real-time implementation of the explicit form of the control law, computed by solving the associated multi-parametric nonlinear problem.**

I. Introduction

Modern combustion engines are multi-input multi-output nonlinear systems with saturated actuators. In this context, the development time and cost of control algorithms have become major issues for car manufacturers. This study is motivated by the desire to tend toward a quasi-systematic engine control design and calibration process. Model predictive control (MPC) in general has become the accepted approach in various industries for controlling multivariable processes [1]. Until now, online computation complexity considerations have prevented it from penetrating the automotive industry [2-4]. This drawback can be circumvented using an explicit approach where an approximation of the solution is computed using multi-parametric programming. This methodology has considerably enlarged the range of possible applications [5-8] and is applied here, to the engine air path control of a turbocharged engine [2, 9, 10].

In this paper, we present the synthesis of an explicit nonlinear predictive controller (NMPC) which relies on a control-oriented physics-based engine model [11]. The objective function is designed to track an inlet manifold pressure set point while optimizing the engine efficiency. The explicit approach leads to a piecewise affine control law which, once stored in a binary search tree, matches the requirement for real-time implementation, i.e. provides both fast and predictable computation time. The conclusion stresses the fact that NMPC combined with the explicit approach is a major step toward a systematic engine control design and calibration process.

J. El Hadef is with the University of Orleans, Laboratoire PRISME, 8 rue Leonard de Vinci, 45000 Orleans, FRANCE and Renault SA, CTL, 1 allée de Cornuel, 91510 Lardy, FRANCE (e-mail: jamil.el-hadef@etu.univ-orleans.fr).

S. Olaru and P. Rodriguez-Ayerbe are with the E3S (Supelec System Science), Automatic Control Department, FRANCE (e-mail: sorin.olaru@supelec.fr).

G. Colin and Y. Chamaillard are with the University of Orleans.

V. Talon is with Renault SA (e-mail: vincent.talon@renault.com).

The paper is organized as follows. Section II presents the system and the control objectives. Section III presents insights into the physics-based model of the system. The NMPC law is presented in section IV while the algorithm used to compute the explicit control law is detailed in section V. Finally, comparative simulation results are presented in the last section.

II. System Description and Control Objectives

The objective is to design a control law for the air path of a downsized turbocharged spark-ignited (SI) engine (Fig. 1).

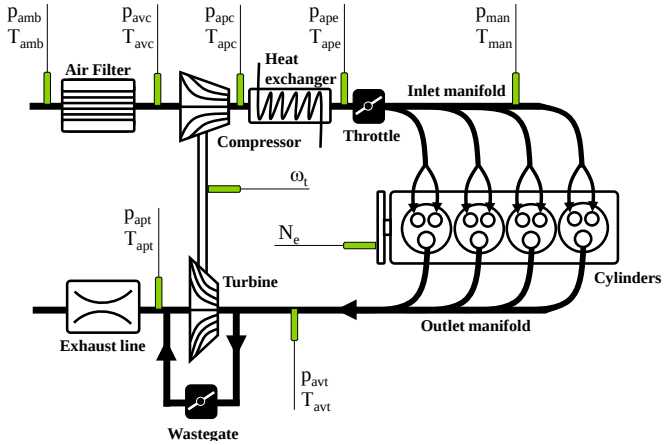

Figure 1. Air path sketch of a turbocharged SI engine (p stands for pressure, T for temperature, N_e and ω_t are respectively the engine and turbocharger rotational speed).

At the intake (p_{avc},T_{avc}), a compressor and a heat exchanger successively increase the pressure (p_{apc},T_{apc}) and cool down the fresh air flow (p_{ape},T_{ape}). Then, a variable flow restriction, called throttle, controls the inlet manifold pressure p_{man}. At the exhaust (p_{avt},T_{avt}), the amount of gas which passes through a turbine is controlled by a by-pass, known as a wastegate (Fig. 1). This energy, recovered at the exhaust, drives the intake compressor and its outlet boost pressure p_{apc} through a shaft (ω_t) [12]. This introduces a physical feedback path in the engine.

In order to maximize the efficiency of the three-way catalytic converter, SI engines operate with an air/fuel equivalent ratio of one. Consequently, from the control point of view, the engine torque is directly controlled by the air mass entering the cylinders. For a given engine speed, this mass directly depends on the inlet manifold pressure and temperature. A supervisor usually provides the inlet manifold pressure reference trajectories from the engine torque that must be achieved. Thus, the air path controller task is to determine which throttle and wastegate positions will allow a

given inlet manifold set point to be reached in the minimum amount of time. Pollutant emissions and drivability consideration are taken into account by the supervisor.

III. Physics-Based 0D Engine Model

When considering real-time automotive model-based control, authors usually use multiple piece-wise linear or linearized models to describe the nonlinear behavior of combustion engines [13]. These models allow fast output prediction but their calibration requires numerous test bench measurements.

In explicit approaches, the model is only used offline, in a multi-parametric program [5, 7]. For this reason, we propose to use a single physics-based nonlinear model. In order to bring the calibration effort under control, the combination of a 0D modeling approach and a mean value cylinder model was chosen [14]. The philosophy behind the model is succinctly described below. For the complete derivation of the model see [11, 12, 15].

A. Hypothesis and Modeling Philosophy

The air path is discretized: a control volume of the air path is followed by a flow restriction, itself followed by another control volume and so forth (Fig. 2).

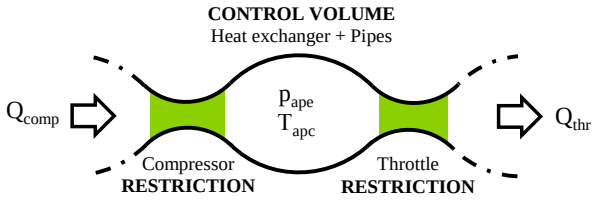

Figure 2. Example of a succession of control volumes and restrictions: the heat exchanger and its pipes are surrounded by two flow restrictions: the compressor and the throttle.

In each control volume, the pressure and the temperature describe the complete thermodynamic state of the volume. In each of them, the pressure dynamics is described by a differential equation deduced from Euler's mass, energy and momentum equations [11]. The temperature dynamics can be neglected and computed through algebraic relations [16, 17]. Altogether, the model contains three control volumes: the inlet and outlet manifolds and the heat exchanger (Fig. 1). It respectively corresponds to three states: p_{man}, p_{avt} and p_{ape}. A fourth state describes the turbocharger rotational speed ω_t.

B. Pressures in the Three Control Volume

In a given control volume V (Fig. 2), the pressure time derivative $\dot{p}$ is given by:

$$\dot{p} = \frac{\gamma r}{V}\left(Q_{m_{in}} T_{in} - Q_{m_{out}} T_{out}\right) \quad (1)$$

where γ is the ratio of specific heat, r is the fluid gas constant, T the flow temperature and Q_m the mass flow rate. Indices "in" and "out" respectively stand for inlet and outlet of the considered control volume.

C. Turbocharger Model

The turbocharger rotational speed ω_t is given by:

$$\dot{\omega_t} = \frac{1}{I}\left(\Gamma_{turb} - \Gamma_{comp}\right) \quad (2)$$

where I is the turbocharger inertia of the shaft which links the turbine to the compressor and Γ_{turb} and Γ_{comp} respectively represent the turbine and compressor torques.

D. Actuator Models

The system has two inputs: the throttle and the wastegate positions, respectively u_{pap} and u_{wg}. Both actuators are considered as variable flow restrictions. The flow is computed using pressures on each side:

$$\begin{cases} Q(u) = \frac{p_{us}}{\sqrt{rT_{us}}} A_{eff}(u) \Pi^{\frac{1}{\gamma}} \sqrt{\frac{2\gamma}{\gamma-1}\left(1 - \Pi^{\frac{\gamma-1}{\gamma}}\right)} \\ \Pi = \max\left(\frac{p_{ds}}{p_{us}}, \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma}{\gamma-1}}\right) \end{cases} \quad (3)$$

where A_{eff} is the effective area of the orifice and depends nonlinearly on the actuators' position: $u = u_{pap}$ or $u = u_{wg}$. The indices "us" and "ds" respectively stand for upstream and downstream.

As explained in [16], the dynamics of the actuators are neglected.

E. Summary

The physics-based model is nonlinear and has four states: p_{apc}, p_{man}, p_{avt} and ω_t. There are two control variables: u_{pap} and u_{wg}. In continuous time, it is written as below:

$$\begin{cases} \dot{p_{ape}} = \frac{\gamma r}{V_{ape}}\left(Q_{comp} T_{apc} - Q_{thr}(u_{pap}) T_{ape}\right) \\ \dot{p_{man}} = \frac{\gamma r}{V_{man}} T_{man}\left(Q_{thr}(u_{pap}) - Q_{eng}\right) \\ \dot{p_{avt}} = \frac{\gamma r}{V_{avt}} T_{avt}\left(Q_{eng} + Q_{fuel} - Q_{turb} - Q_{wg}(u_{wg})\right) \\ \dot{\omega_t} = \frac{1}{I}\left(\Gamma_{turb} - \Gamma_{comp}\right) \end{cases} \quad (4)$$

where V_{ape}, V_{man} and V_{avt} respectively represent the volume between the compressor and the throttle, the volume of the intake manifold and the exhaust manifold volume (Fig. 1). Q_{eng} is the engine mass air flow and Q_{fuel} the injected fuel quantity. Q_{thr} and Q_{wg} stand for the throttle and wastegate flows, both obtained with (3).

For control design purposes, the model is discretized at a sampling time of 1 ms, using Euler's backward differentiation method. Results concerning steady-state and transient validation of the discrete-time model are given in [15]. The model validation stage shows that the prediction error remains well below 10% for steady-state operating points as well as transients.

IV. Nonlinear Model Predictive Control

A. NMPC Formulation for Reference Tracking

MPC usually uses an iterative finite-time open loop optimization to compute the optimal actuator position vector u^* with respect to an objective function S. At each time step, only the first command is applied to the real process. A new open-loop optimal problem is solved following the receding horizon principle [1, 2, 18].

Given the current system state x_0 and the vector of exogenous inputs σ (principally the set points) at time instant k, the discrete-time NMPC problem can be written as below:

$$S^* = S(u^*) = \min_{u(.)} \left[S = \sum_{i=k}^{k+N_p} J(x(i), y(i), u(i), \sigma) \right] \quad (5)$$

s.t
$$x(k+1) = f(x(k), u(k), \sigma) \quad (6)$$
$$y(k) = g(x(k), u(k), \sigma) \quad (7)$$
$$\underline{x} \leq x(k) \leq \overline{x} \quad (8)$$
$$\underline{u} \leq u(k) \leq \overline{u} \quad (9)$$
$$x(k) = x_0 \quad (10)$$

where $x(i)$ denotes the system states and $u(i)$ stands for the vector of piecewise constant control inputs. J is called cost function and $H_p = [k, k+N_p]$ is the so-called prediction horizon at time k. f and g are nonlinear functions describing the discrete-time system dynamics. Finally, $\underline{x}$, $\overline{x}$, $\underline{u}$ and $\overline{u}$ respectively stand for lower and upper bounds on the states and the control variables.

B. Application to a Turbocharged SI Engine

The air path control problem fits into the above formulation of an NMPC optimal control problem if we consider:

$$x := \left(p_{ape}, p_{man}, p_{avt}, \omega_t\right)^T \quad (11)$$
$$y := (p_{man}, p_{avt}) \quad (12)$$
$$u := \left(u_{pap}, u_{wg}\right) \quad (13)$$
$$\sigma = (N_e, p_{man}^{SP}, p_{amb}, T_{amb}) \quad (14)$$

where T_{amb} is the ambient temperature and p_{man}^{SP} is the pressure set point, used to track the desired manifold pressure set point and is assumed to be constant over the prediction horizon. The right hand side of (6) is derived from the discrete-time version of the set of equations (4), while the output vector y consists of the second and third states of the same model. No explicit state constraints are imposed. The manipulated variables are bounded in order to take into account actual actuator saturations:

$$0 \leq u \leq 100\% \quad (15)$$

Since there is an infinite number of combinations of actuator positions to achieve a given inlet manifold pressure, one would like to select the most efficient one. The efficiency on an engine cycle can be computed from the integral of the cylinder pressure-volume (p-V) diagram of four-stroke engines [12]. In particular, the pumping losses are directly linked to the difference between the exhaust manifold pressure p_{avt} and the inlet manifold pressure p_{man}. In order to reduce the pumping losses, i.e. increase the engine efficiency, this thermodynamic consideration is directly taken into account in the objective function [16]. A first term ensures set point tracking while a second one minimizes the pressure ratio between exhaust and inlet manifold:

$$S = \sum_{i=k}^{k+N_p} \left[\alpha \left(p_{man}^{SP} - p_{man}(i) \right)^2 + \beta \frac{p_{avt}(i)}{p_{man}(i)} \right] \quad (16)$$

where the weighting factors α and β are used to scale and penalize each term of the cost function. If α is chosen too small compared to β, the inlet manifold pressure will not reach its desired value.

The NMPC formulation (5-10) avoids the use of a terminal penalty term and terminal constraints [19]. This design choice is principally due to the parameterization of the optimization problem in σ which makes the definition of such additional terms impractical in this nonlinear framework. However, the closed-loop stability (in the sense of boundedness of state space trajectories) is ensured by the intrinsic dissipative properties of the system as well as the bounds on the control variables.

The only thing left is a proper choice of the control sampling time: in order to control the fast dynamics of the engine as well as allow fast response to set point changes a new control is applied every 10 ms.

C. Prediction Horizon and Control Variable Partition

In this study, a prediction and control horizon of 100 ms combined with a constant control value over these horizons proved to be flexible enough to track realistic vehicle transients [16]. The optimization problem is addressed in a classical single shooting framework using the trust-region reflective method implemented in Matlab®. This algorithm, initialized in the middle of the control variable space $u = 50\%$, converges within 10 iterations to a close-to-global solution. Results are depicted in section VI and in [16].

The characteristics of the problem (5)-(10), i.e. the small number of model states and set points, as well as its short convergence time match the requirements for computing an explicit solution. In particular, (5)-(10) can be considered as a multi-parametric nonlinear program (mp-NLP) since it is a nonlinear program in u parameterized by x and σ. The next section provides insights in the synthesis of the piecewise affine (PWA) solution to this mp-NLP and its real-time implementation.

V. Explicit Nonlinear Model Predictive Control

The PWA approximation of the solution of (5) is computed using mp-NLP [7, 20]. The constituent functions will be defined on hyper-rectangles, which enables the parameter partition to be represented as a binary search tree [21]. Online, the time-consuming MPC optimization is then replaced by positioning via a simple search tree. This positioning mechanism fulfils all the criteria for a real-time implementation: i.e. both fast and predictable calculation time. During the synthesis, the accuracy of approximation is quantified by the difference between the close-to-global and sub-optimal objective function values.

In the sub-sections below, we successively detail the different steps to build the PWA law from the NMPC algorithm presented in section IV. The complete algorithm includes: generating a set of points, computing the affine approximation and then, its validation on another set of points. Finally, if needed, the hyper-rectangle is split in order to refine the affine approximation, based on a direct evaluation based rule as presented below.

A. Splitting Rule (Procedure 1)

Suppose X to be the complete parameter space to be explored. The split of a given hyper-rectangle $X_j \subseteq X \subseteq \mathbb{R}^n$ is obtained by splitting the hyper-rectangle on all parameter

space axes by hyper-planes through its center θ_0^j. As a result of the splitting step, 2^n new hyper-rectangles are generated (Fig. 3).

B. Set of Points Generation (Procedure 2)

For a given hyper-rectangle $X_j \subseteq X \subseteq \mathbb{R}^n$ we note $\Theta^j = \left\{\theta_1^j, \theta_2^j, \dots, \theta_{N_\theta}^j\right\}$ its $N_\theta = 2^n$ vertices (Fig. 3). This set of points is used to compute the approximated control law. We note $\Phi^j = \left\{\varphi_1^j, \varphi_2^j, \dots, \varphi_{N_\varphi}^j\right\}$ the union of all the vertices of the hyper-rectangles that would be created by using the splitting rule above (Fig. 3). This set of N_φ vertices is used to validate the approximated control law. It can be seen that $\Theta^j \subset \Phi^j$.

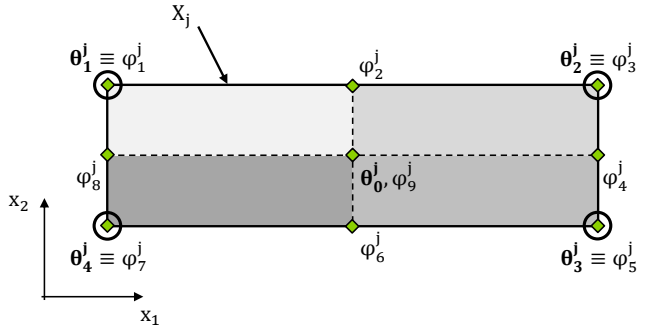

Figure 3. Illustration of the splitting rules (Procedure 1) in $n = 2$ dimension parameter space: X_j is split on both parameter space axes through its center θ_0^j. It generates 2^n new hyper-rectangles. The sets of points generated using Procedure 2 are also displayed: X_j has 2^n vertices $\Theta^j = \left\{\theta_1^j, \theta_2^j, \dots, \theta_{N_\theta}^j\right\}$ (black circles) which are used to compute the approximated control law. The control law is then validated on the set of points $\Phi^j = \left\{\varphi_1^j, \varphi_2^j, \dots, \varphi_{N_\varphi}^j\right\}$, displayed using green diamonds. They correspond to the vertices of the 2^n new hyper-rectangles.

C. Computing the Affine Approximated Control Law $\hat{u}_{X_j} = K_j x + h_j$ (Procedure 3)

Consider any hyper-rectangle $X_j \subseteq X \subseteq \mathbb{R}^n$ and its vertices $\Theta^j = \left\{\theta_1^j, \theta_2^j, \dots, \theta_{N_\theta}^j\right\}$ as defined above. The parameters K_j and h_j of the close-to-global solution approximation are obtained by linear regression analysis on the vector of u^*, the close-to-global solutions at the N_θ vertices of X_j. The advantage of this method compared to approximating the close-to-global solution S^* of the mp-NLP, as presented in [7, 20], is that it requires less computational time by avoiding running the model iteratively. However, as in [7, 20], the error between the optimal and sub-optimal objective function values remains the ultimate criterion to validate the approximated control law.

D. Error Bounds Approximation (Procedure 4)

Consider a hyper-rectangle $X_j \subseteq X \subseteq \mathbb{R}^n$ and its associated validation set of points Φ^j. The estimated error bound $\hat{\varepsilon}_j$ of the error bound ε_j is taken as the maximum estimated error bound obtained on the N_φ validation points Φ^j:

$$\hat{\varepsilon}_0 = \max_{i \in \{1,2,\dots,N_\varphi\}} \left(\hat{S}_{\varphi_i^j} - S^*_{\varphi_i^j}\right) \quad (17)$$

where $\hat{S}$ is the sub-optimal objective function value, obtained with the approximated affine control law.

E. Algorithm

For a given hyper-rectangle $X_j \subseteq X \subseteq \mathbb{R}^n$, we note Δ_x the vector of lengths of the hyper-rectangle along x. Assume that the maximal tolerance $\bar{\varepsilon}_0 > 0$ of the objective function approximation error is given as well as the vector of the minimal hyper-rectangle allowed: $\underline{\Delta_x} > 0$. The explicit approximate PWA control law is computed using the algorithm below:

Input: The first hyper-rectangle X_0 which represents the entire states space to be explored. The maximum approximation tolerance $\bar{\varepsilon}$ and the vector of minimal allowed lengths of the hyper-rectangle $\underline{\Delta_x}$. We assume $\Delta_{x_0} > \underline{\Delta_x}$.

Output: Partition $\Pi = \left\{X_1, X_2, \dots, X_{N_X}\right\}$ and associated PWA control law $\hat{u}_\Pi = \left\{\hat{u}_{X_1}, \hat{u}_{X_2}, \dots, \hat{u}_{X_{N_X}}\right\}$.

1. Initialize the partition to the first hyper-rectangle, i.e. $\Pi = X_0$. Move the hyper-rectangle X_0 to List 1: the list of hyper-rectangles to be explored.
2. Compute Θ^0 the set of vertices of X_0 as defined in Procedure 2.
3. Compute a close-to-global solution $u^*_{X_0}(\theta_i^0)$ to problem (5)-(10) for each vertices $i \in \{1,2,\dots,N_\theta\}$ of X_0, i.e. every point of Θ^0.
4. **while** List 1 is not empty **do**
5. Select the first unexplored hyper-rectangle X_j in List 1. Thanks to previous calculations, its set of vertices Θ^j and the associated close-to-global solution $u^*_{X_j}(\theta_i^j)$ to (5)-(10) are available. This is also the case for the corresponding close-to-global objective function value: $S^*_{\theta_i^j}$.
6. Compute the approximated affine control law in the hyper-rectangle X_j, i.e. $\hat{u}_{X_j} = K_j{}^T x + h_j$ as detailed in Procedure 3.
7. Split the hyper-rectangle X_j as detailed in Procedure 1. Store the hyper-rectangles X_k, $k \in \{1,2,\dots,2^n\}$ obtained in List 2 : a temporary list of hyper-rectangles.
8. Compute the vertices of each hyper-rectangle contained in List 2 in order to obtain Φ^j as defined in Procedure 2. This list will provide the validation set of points of X_j.
9. Compute a close-to-global solution $u^*_{X_j}(\varphi_i^j)$ to problem (2)-(10) at each vertex contained in the set Φ^j. Store the corresponding close-to-global objective function value $S^*_{\varphi_i^j}$.
10. Enhance List 2 by storing the close-to-global solution computed at step 9 $u^*_{X_k}(\varphi_i^k)$ for each hyper-rectangle X_k. Do the same for the set of vertices Θ^k of each hyper-rectangle and the corresponding objective function value $S^*_{\theta_i^k}$.
11. Compute the approximate objective function $\hat{S}_{\varphi_i^j}$ at

each vertex contained in Φ^j, using the affine control law obtained in step 6.

12. Compute the estimate of the error bound $\hat{\varepsilon}_0$ in X_j, using the set of points Φ^j as specified in Procedure 4, the approximate solution computed at step 11 and the close-to-global solution value from step 9.
13. Remove the hyper-rectangle X_j from List 1
14. **if** $\hat{\varepsilon}_0 < \bar{\varepsilon}$ **then**
15. Save the hyper-rectangle X_j, its vertices Θ^j and the approximated affine control law $\hat{u}_{X_j} = K_j{}^T x + h_j$ obtained at step 6. Mark X_j as validated w.r.t the error bound criterion $\bar{\varepsilon}$ and the sensitivity level Δ_x.
16. **else**
17. **if** $\frac{1}{2}\Delta_x > \underline{\Delta_x}$ **then**
18. Add the hyper-rectangles contained in List 2 at the top of List 1. Clear List 2.
19. **else**
20. Save the hyper-rectangle X_j, its vertices Θ^j and the approximated affine control law $\hat{u}_{X_j} = K_j{}^T x + h_j$ obtained in step 6. Mark X_j as validated w.r.t the sensitivity level Δ_x only.
21. **end if**
22. **end if**
23. **end while**

This algorithm guarantees that a PWA feedback control law with a finite optimal number of hyper-rectangular regions will be obtained. The maximum number of elements in the partitions is a quantitative criterion of the exploration, controlled by the vector of maximum lengths along x_i of the hyper-rectangle $\underline{\Delta_{x_i}}$ at step 20 of the algorithm. The effective number of elements is controlled at step 14 by the pre-imposed qualitative criterion: the error bound $\bar{\varepsilon}$. In our particular case, the maximum number of hyper-rectangles is greater than 2,000,000. The algorithm presented above leads to a final partition of 3917 elements, thus providing the adaptation capabilities of the proposed procedure. The splitting operation is used only when the nonlinear characteristics of the dynamics impose it.

F. PWA Control Law Evaluation

The online computation time of a PWA control law is directly linked to the complexity of the representation, i.e. the number of polyhedrons M. Strategies for reducing the number of regions exist [7]. Joining convex unions of polyhedrons which share the same affine control law is one of them.

Another crucial element for an efficient and fast online implementation relies on the approach that is used to read the PWA function. It is a truism that evaluating one by one each region is computationally unsustainable. A more efficient alternative, proposed in [21] relies on a binary search tree representation of the partition. This method is particularly suitable in our case, where the partition relies on hyper-rectangles. The logarithmic complexity in the number of regions considerably reduces the computation time with respect to direct approaches, even considering parallel implementation.

In this study, a binary search tree is built using the MPT toolbox [22]. This allows embedding a table of M rows and $n + 3$ columns. As for the evaluation of the PWA control law it consists of a two steps procedure [23] :

1. **Searching the tree:** starting at the root node and then at each step, a hyperplane cut $c_j^T x + d_j$ is evaluated. The child node is chosen according to the sign of this expression. This step is repeated until a leaf node is reached.
2. **PWA control evaluation**: when the leaf node j has been reached, the control is directly given by evaluating $\hat{u}_{X_j} = c_j^T x + d_j$.

At each node the arithmetic operations required are n multiplications, n additions and one comparison. An estimate of the search tree depth D is usually given by $D = 1.7 \log_2 M$.

VI. Control Performances Simulation

For the standard NMPC scheme (using iterative online optimization), performances on the MVEG (Motor Vehicle Emissions Group) driving cycle are depicted on Fig. 4. This cycle is designed to assess the fuel consumption and pollutants emissions of car engines in Europe and as such represent relevant engine operating conditions. It can be seen that the pressure set point tracking performances are acceptable: inlet manifold pressure tracking error remains under 50 mbar. 100 mbar represents a rough estimate of the maximum tracking error required to ensure good drivability.

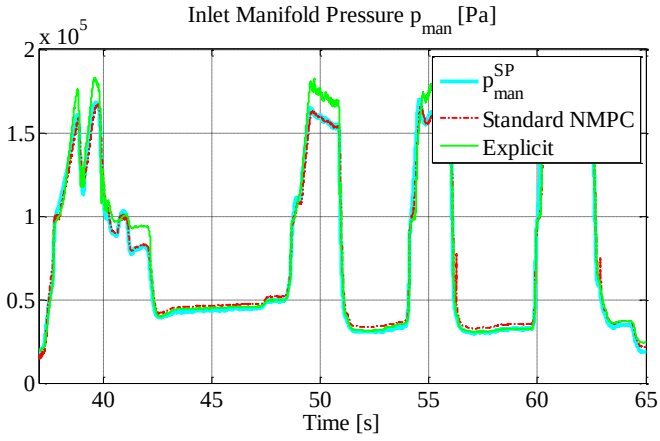

Figure 4. Inlet manifold pressure tracking performances for different NMPC schemes are depicted on the first graph. The standard NMPC uses iterative online optimization and the explicit NMPC is computed using the algorithm presented in the paper.

The mp-NLP problem has 4 constraints (15) and n = 6 free variables:

$$\theta_i^j = \left(p_{ape}, p_{man}, p_{avt}, \omega_t, p_{man}^{SP}, N_e\right)^T \qquad (18)$$

This leads to a partition of 3,917 hyper-rectangles stored in a binary search tree that approximately requires 71 kilobytes of memory. That represents about 2.5% of the memory available in current standard Engine Control Units (ECU). In this particular example, the table that must be embedded contains 35,253 elements. This is in agreement with current 16 or 32 bits ECU that can respectively handle tables of $2^{16}-1$ and $2^{32}-1$ elements.

The performances of the explicit controller are presented on Fig. 4 and Fig. 5. These performances are obtained by directly applying the actuator affine control laws stored in the binary search tree. On a Core™ i7 CPU x8 cores, without any specific Matlab® code optimization, the average computation time of the control law is reduced by a factor of 10,000 with respect to the online optimization used in the standard NMPC. The average computation time of the control inputs is 0.09 ms with a standard deviation of about 0.05 ms, i.e. far below the engine control sampling time: 10 ms. With such a low computation time, the implementation of the control law on an actual ECU becomes worth considering. In fact, in this particular example, the number of arithmetic operations required to compute the control is in perfect agreement with current ECU performances: 72 multiplications, 72 additions and 12 comparisons at each time step.

However, since no integral actions have yet been considered, the tracking performances depicted on Fig. 4 and Fig. 5 are not acceptable. For example between 41 and 42 seconds, a steady-state error of more than 100 mbar is encountered. One way to eliminate this offset is to introduce an integral action directly in the parametric control design by adding an integral state [24]. A drawback of this method is that it increases the number of states of the model, i.e. the number of parameters of the mp-NLP. The controller complexity could then become impractical.

In order to circumvent this problem we propose to enhance the control scheme based on the explicit approach by adding an integral action at the end of the synthesis. Because of the efficient feed forward action achieved by the explicit NMPC, a single integral action, acting on the throttle, is required.

The control actions are then computed as below:

$$\begin{pmatrix} \hat{u}^*_{pap} \\ \hat{u}^*_{wg} \end{pmatrix} = K_j{}^T x + h_j + \begin{pmatrix} K_{integral} \\ 0 \end{pmatrix} \int (p^{SP}_{man} - p_{man}) \quad (19)$$

where $K_{integral}$ is constant through the entire operating range and calibrated using a trial-and-error calibration method. In order to compensate with typical integral-windup effect, an anti-windup integral controller was implemented. The integral action must be sufficiently slow not to interfere significantly with the explicit control law during transients. [16]

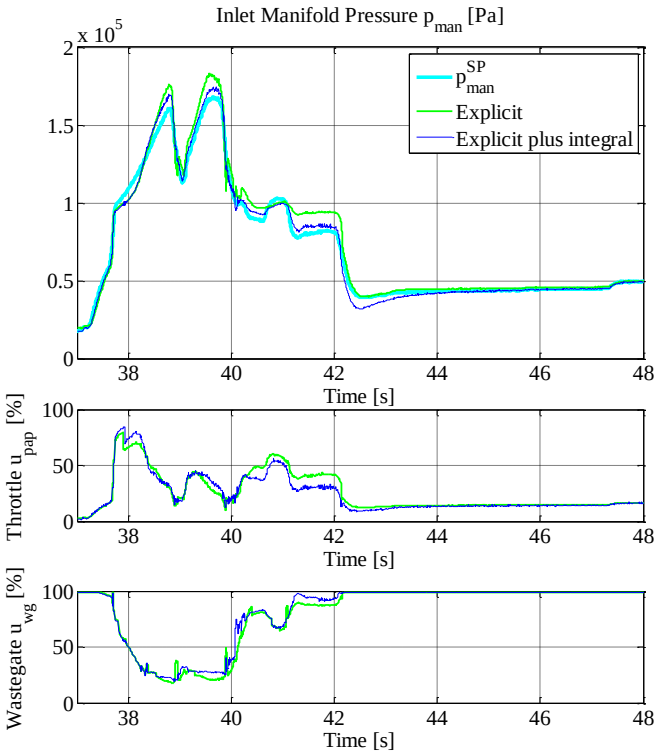

Figure 5. Inlet manifold pressure tracking performances for two explicit NMPC schemes : with and without anti-windup integral action on the throttle. Comparative actuators position graphs are depicted.

VII. Conclusion

The results shown in this paper confirm the fact that NMPC is predetermined to handle complex constrained nonlinear systems. It also demonstrates that the explicit approach is suitable to bring a NMPC control law to a real time implementation while matching automotive industry constraints in terms of computation time. The price to pay is an increase in memory requirements due to the PWA control law storage. In this approach, the use of a physics-based model will ensure a robust exploration of the parameter space of the mp-NLP. This approach is currently being extended to more complex technical definitions where more significant performance enhancement can be achieved. In fact, since efficiency criteria can directly be added to the objective function, one can guarantee to maximize the use of a given engine layout.

Another major issue relies on the step that has been accomplished toward a quasi-systematic approach for designing engine air path control strategies. In fact, the combination of a physics-based model with the explicit NMPC scheme leads to an extreme reduction in tuning work. The nonlinear, constrained and coupled nature of the system is implicitly taken into account.

Further extensions are under way in order to validate experimentally the controller performances as well as assess the overall tradeoff between memory and online computational requirements.

REFERENCES

[1] E. F. Camacho and C. Bordons, *Model Predictive Control*, 2 ed. London: Springer, 2004.

[2] L. Del Re, L. Glielmo, C. Guardiola, and I. Kolmanovsky, *Automotive Model Predictive Control*. Berlin: Springer, 2010.

[3] M. A. Henson, "Nonlinear Model Predictive Control: Current Status and Future Directions," *Computers & Chemical Engineering*, vol. 23, pp. 187-202, 1998.

[4] M. Herceg, T. Raff, R. Findeisen, and F. Allgowe, "Nonlinear Model Predictive Control of a Turbocharged Diesel Engine," in *IEEE International Conference on Control*, 2006, pp. 2766-2771.

[5] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The Explicit Solution of Model Predictive Control via Multiparametric Quadratic Programming," in *American Control Conference*, Chicago, 2000.

[6] S. Di Cairano, A. Bemporad, I. Kolmanovsky, and D. Hrovat, "Model Predictive Control of Magnetically Actuated Mass Spring Dampers for Automotive Applications," *Int. J. of Control*, vol. 80, pp. 1701-1716, 2007.

[7] A. Grancharova and T. A. Johansen, *Explicit Nonlinear Model Predictive Control*. Berlin: Springer, 2012.

[8] F. Borrelli, M. Baotic, J. Pekar, and G. Stewart, "On the Complexity of Explicit MPC Laws," in *Proceedings of European Control Conference 2009*, Budapest, Hungary, 2009.

[9] J. Deng, R. Stobart, C. Liu, and E. Winward, "Explicit Model Predictive Control of the Diesel Engine Fuel Path," *SAE Technical Paper*, 2012.

[10] G. Stewart and F. Borrelli, "A Model Predictive Control Framework for Industrial Turbodiesel Engine Control," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, 2008, pp. 5704-5711.

[11] L. Eriksson, "Modeling and Control of Turbocharged SI and DI Engines," *Oil & Gas Science and Technology - Rev. IFP Energies nouvelles*, vol. 62, pp. 523-538, 2007.

[12] J. B. Heywood, *Internal Combustion Engines Fundamentals*: McGraw-Hill, 1988.

[13] J. Pekar, P. Garimella, D. Germann, and G. E. Stewart, "Experimental Results for Sensor Selection and Multivariable Controller Design for a Heavy-Duty Diesel Engine," presented at the IFAC Workshop on Engine and Powertrain Control, Simulation and Modeling, France, 2012.

[14] P. Moulin, J. Chauvin, and B. Youssef, "Modelling and Control of the Air System of a Turbocharged Gasoline Engine," *Proc. of the IFAC World Conference*, 2008.

[15] J. El Hadef, G. Colin, Y. Chamaillard, and V. Talon, "Turbocharged SI Engine Models for Control," in *The 11th International Symposium on Advanced Vehicle Control - AVEC'12*, Seoul, Korea, 2012.

[16] J. El Hadef, G. Colin, Y. Chamaillard, S. Olaru, P. Rodriguez-Ayerbe, and V. Talon, "Explicit-Ready Nonlinear Model Predictive Control of the Air Path of a Turbocharged Spark-Ignited Engine," in *7th IFAC Symposium on Advances in Automotive Control*, Tokyo, Japan, 2013.

[17] E. Hendricks, "Isothermal versus Adiabatic Mean Value SI Engine Models," *3rd IFAC Workshop, Advances in Automotive Control*, pp. 373-378, 2001.

[18] H. J. Ferreau, G. Lorini, and M. Diehl, "Fast Nonlinear Model Predictive Control of Gasoline Engine," in *International Conference on Control Applications*, Munich, Germany, 2006.

[19] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and S. P.O.M., "Constrained Model Predictive Control : Stability and Optimality," *Automatica*, vol. 26, 2000.

[20] T. A. Johansen, "On Multi-Parametric Nonlinear Programming and Explicit Nonlinear Model Predictive Control," in *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, 2002, pp. 2768-2773 vol.3.

[21] P. Tøndel, T. A. Johansen, and A. Bemporad, "Evaluation of Piecewise Affine Control via Binary Search Tree," *Automatica*, vol. 39, pp. 945-950, 2003.

[22] M. Kvasnica, *Real-Time Model Predictive Control via Multi-Parametric Programming: Theory and Tools*: VDM Verlag, 2009.

[23] T. A. Johansen, W. Jackson, R. Schreiber, and P. Tøndel, "Hardware Synthesis of Explicit Model Predictive Controllers," *IEEE Transactions on Control Systems Technology*, vol. 15, pp. 191-197, 2007.

[24] P. L. Lee and G. R. Sullivan, "Generic Model Control (GMC)," *Computers & Chemical Engineering*, vol. 12, pp. 573-580, 1988.